AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application.

Claims 1-35 (Cancelled)

36. (New) A method comprising:

compiling source code written in a high level language into an intermediate representation;

compiling the intermediate representation into an executable binary;

validating the executable binary;

modifying the source code based on the validation of the executable binary; and

recompiling the modified source code to generate a new intermediate representation.

37. (New) The method of claim 36, wherein compiling the source code further comprises inserting one or more annotations into the intermediate representation.

38. (New) The method of claim 36, wherein said compiling the intermediate representation comprises generating code to perform one or more tests, and wherein said validating the executable binary comprises performing the one or more tests.

39. (New) The method of claim 38, wherein the one or more tests comprise one or more tests selected from the group consisting of checks for use of uninitialized variables, checks for

bad pointers before dereferences, and array bounds checks prior to making array references.

40. (New) The method of claim 36, further comprising distributing the new intermediate representation to an end user.

41. (New) A computer readable medium having stored thereon data representing instructions that when executed cause a processor to:

compile source code written in a high level language to an intermediate representation;

compile the intermediate representation to an executable binary;

validate the executable binary; and

recompile source code that has been modified based on the validation of the executable binary to generate a new intermediate representation.

42. (New) The computer readable medium of claim 41, wherein the data further comprises data representing instructions that when executed cause the processor to:

insert one or more annotations into the intermediate representation.

43. (New) The computer readable medium of claim 41, wherein the data further comprises data representing instructions that when executed cause the processor to:

generate code to perform one or more tests; and

perform the one or more tests.

44. (New) A method comprising:

running executable code;

collecting profile data while the executable code is running;

when the CPU is idle, processing the profile data; and

recompiling software based on the processed profile data.

45. (New) The method of claim 44, wherein said processing the profile data comprises generating one or more profiles.

46. (New) The method of claim 44, wherein said generating the one or more profiles comprises:

generation of a binary level profile from analysis of the profile data; and

derivation of a profile at high level intermediate language from the binary level profile.

47. (New) A computer readable medium having stored thereon data representing instructions that when executed cause a processor to:

run executable code;

collect profile data while the executable code is running;

when the CPU is idle, process the profile data; and

recompile software based on the processed profile data.

48. (New) The computer readable medium of claim 47, wherein the data further comprises data representing instructions that when executed cause the processor to:

generate one or more profiles.

49. (New) The computer readable medium of claim 47, wherein the data further comprises data representing instructions that when executed cause the processor to:

generate a binary level profile from analysis of the profile data; and

derive a profile at high level intermediate language from the binary level profile.

50. (New) A method comprising:

compiling software; and

including a compilation annotation in the compiled software.

51. (New) The method of claim 50, further comprising including an annotation in the intermediate representation that includes information describing how a binary level instruction of the compiled software evolved from a corresponding high level instruction of source code.

52. (New) The method of claim 50, further comprising using an annotation to map an instruction to a source level token.

53. (New) The method of claim 50, further comprising using an annotation to communicate between phases of a compiler across compilations.

54. (New) The method of claim 50, further comprising recording information by optimization phase in an annotation.

55. (New) The method of claim 50, further comprising creating an annotation by:

creating a new action node;

assigning to the new action node a major ID from a precomputed ID;

assigning a new action number to the new action node;

setting a previous action node pointer of the new action node to NULL;

marking a compiler phase in which the new node was created; and

marking an action of the new node as created.

56. (New) The method of claim 50, further comprising duplicating an annotation by:

creating two new action nodes;

copying a major ID to the new action nodes from an action node of instructions being copied;

assigning new action numbers to the two new nodes;

setting previous action node pointers of the new nodes to the action node being copied;

marking a compiler phase in which the nodes was duplicated; and

marking an action of the new nodes as duplicated.

57. (New) The method of claim 50, further comprising deleting an annotation by:

creating a new action node;

copying a major ID from an action node of an instruction being deleted to the new action node;

assigning a new action number to the new action node;

setting a previous action pointer in the new action node to the action node of the instruction being deleted;

marking a compiler phase in which the node was deleted; and

marking an action of the deleted node as deleted.

58. (New) The method of claim 50, further comprising merging annotations by:

creating a new action node;

copying a major ID from a previous action node of instructions being merged;

assigning a new action number to the new action node;

setting a previous action pointer of the new node to a list of nodes of the instruction being merged;

adding the new action to a next actions pointer list of previous actions;

marking a compiler phase in which the node was merged;

marking an action of the new node as merged.

59. (New) The method of claim 50, further comprising performing annotation branch inversion by:

creating a new action node;

copying a major ID from a branch instruction being inverted;

assigning a new action number to the new node;

setting a previous action pointer of the new node to a node of a branch being inverted;

marking a compiler phase in which the inversion occurred; and

marking the branch as inverted.

60. (New) The method of claim 50, further comprising relating locations in an executable to locations in a profile database and the intermediate representation by using an annotation.

61. (New) A computer readable medium having stored thereon data representing instructions that when executed cause a processor to:

compile software; and

include a compilation annotation in the compiled software.

62. (New) The computer readable medium of claim 61, wherein the data further comprises data representing instructions that when executed cause the processor to:

relate locations in an executable to locations in a profile database and the intermediate representation by using an annotation.

63. (New) The computer readable medium of claim 61, wherein the data further comprises data representing instructions that when executed cause the processor to:

include an annotation in the intermediate representation that includes information describing how a binary level instruction of the compiled software evolved from a corresponding high level instruction of source code.

64. (New) The computer readable medium of claim 61, wherein the data further comprises data representing instructions that when executed cause the processor to:

record information by optimization phase in an annotation.

65. (New) The computer readable medium of claim 61, wherein the data further comprises data representing instructions that when executed cause the processor to create an annotation by:

creating a new action node;

assigning to the new action node a major ID from a precomputed ID;

assigning a new action number to the new action node;

setting a previous action node pointer of the new action node to NULL;

marking a compiler phase in which the new node was created; and

marking an action of the new node as created.

66. (New) A computer system comprising:

a bus;

a communication device coupled with the bus;

a processor coupled with the bus;

a memory coupled with the bus; and

data stored in the memory that represent instructions that when executed cause a processor to:

run executable code;

collect profile data while the executable code is running;

when the CPU is idle, process the profile data; and

recompile software based on the processed profile data.

67. (New) The computer system of claim 66, wherein the data further comprises data representing instructions that when executed cause the processor to:

generate one or more profiles.

68. (New) The computer system of claim 67, wherein the data further comprises data representing instructions that when executed cause the processor to:

generate a binary level profile from analysis of the profile data; and

derive a profile at high level intermediate language from the binary level profile.

69. (New) The computer system of claim 66, wherein the data further comprises data representing instructions that when executed cause the processor to:

create an annotation.